Liverpool John Moores University
School of Computing and Mathematical Sciences

Investigation into Self-Adaptive Software Agents Development

E. Grishikashvili

Distributed Multimedia Systems Engineering Research Group

Technical Report

http://www.cms.livjm.ac.uk/cmsegris/Deliverables/Reports/

This paper provides an overview of Self-Adaptive Software and software agent technology.

27 April 2001

Version: Draft 1.0

Liverpool john Moors University
School of Computing and Mathematical Sciences
Byrom Street, Liverpool, L3 7UF
Tel: + 44 151 231 2263, Fax: Tel: +44 151 231 2263
E-mail: cmsegris@livjm.ac.uk
http://www.cms.livjm.ac.uk/cmsegris

Investigation into Self-Adaptive Software Agents Development

1. Introduction

Society's increasing dependence on software-intensive systems is driving the need for dependable, robust, continuously available systems. The ability to reconfigure a system at run time is one critical aspect of achieving continuously availability. Although operating systems and programming language have provided programmers with the ability to evoke runtime software changes, such mechanisms do not guarantee that a change will have desired effect or maintain application integrity. It is therefore imperative that there are developed approaches to run time system reconfiguration that help us determine what to change, facilitate reasoning about the consequences of a change, and govern change to preserve application integrity. Without this, the risks introduced by runtime reconfiguration may outweigh those associated with shutting down and restarting the system for reconfiguration.

The new approach in area of "Self-Adaptive Software" provides the key for this kind of systems. Self-adaptive software will identify, promote and evaluate new models of code design and run-time support. These new models will allow software to modify its own behavior or order to adapt, at runtime, when exact conditions and inputs are known, to discovered changes in requirements, inputs, and internal external conditions.

…This implies that the software has multiple ways of accomplishing its purpose, and has enough knowledge of its construction to make effective changes at runtime. Such software should include functionality for evaluating its behaviour and performance, as well as the ability to replan and reconfigure its operations in order to improve its operation. Self-adaptive software should also include a set of components for each major function, along with descriptions of the components, so that components of systems also requires the ability to impedance match input/output of sequenced components, and the ability to generate some of this code from specifications. DARPA seek this new basis of adaptation to be applied at runtime, as opposed to development/design time, or as a maintenance activity [Laddaga., 1998].

Self-adaptive software requires high dependability, robustness, adaptability, and availability. From our point of view it will be useful to develop agents with these properties for multi-agent systems. The design of this kind of multi-agent systems requires spatial attention as no self-adaptive agents design has been developed.

Proposed work aims to develop framework for self-adaptive software agent development. Building a good software agent framework is an enormously complex task since it requires a rather complete understanding of a variety of software architecture areas.

In their work “Towards a Viable Reference Architecture for Multi-Agent Supported Holonic Manufacturing Systems” [Laws, 2000] the authors suggest that Viable System Model (VSM) can be regarded as a unifying reference architecture that provides an integrated organizational/software design approach for multi-agent software systems.

The remainder of the paper is organized as follow. The first section describes what is the software agent and gives briefly definitions of agents that are more important for large systems development. The next section presents an overview of current work, the taxonomy of agent and requirements for self-adaptive software agent. Finally, the paper concludes with the future development opportunities.

2. What is the Software Agent?

Before we explain main objectives of our future work we want to define what the software agent is. During the years, many researchers have come up with different definitions of the term.

According the Wooldridge and Jennings definition a software agent is *"... a hardware or (more usually) software-based computer system that enjoys the following properties:*

- *autonomy: agents operate without the direct intervention of humans or others, and have some kind of control over their actions and internal state;*
- *social ability: agents interact with other agents (and possibly humans) via some kind of agent-communication language;*
- *reactivity: agents perceive their environment, (which may be the physical world, a user via a graphical user interface, a collection of other agents, the INTERNET, or perhaps all of these combined), and respond in a timely fashion to changes that occur in it;*
- *pro-activeness: agents do not simply act in response to their environment, they are able to exhibit goal-directed behaviour by taking the initiative."*

By this definition an agent is a rather complex piece of software that must exhibit some rather extraordinary characteristics in order to be called an agent. In practice most agents have these properties, but most do not take each of them to their extreme. Instead, they focus on one or a few.

In his paper entitled “Software Agents: An Overview”, Hyancinth Hwana takes the “Wooldridge Jennings” definition and hones it down to three behavioural attributes, any two of which must be possessed by a software agents. These are (Nwana 1996):

- **Autonomy** – “refers to the principle that agents can operate on their own without the need for human guidance, even though this would sometimes be invaluable. Hence agents have individual internal states and goals, and they act in such a manner as to meet its goals on behalf of its user. A key element of their autonomy is their proactiveness, i.e. their ability to ‘take the initiative’ rather than acting simply in response to their environment. [M. Wooldridge & Jennings, 1995]”
- **Cooperation** – “cooperation with other agents is paramount: it is raison for having multiple agents in the first place in contrast to having just one. In order to

cooperate, agents need to possess a social ability, i.e. the ability to interact with other agents and possibly human via some communication language (Wooldridge & Jennings, 1995).” Having said this, it is possible for agents to coordinate their actions without cooperation (Nwana et al. 1996).

- **Learning** – “For agent systems to be truly ‘smart’, they would have to learn as they react and/or interact with their external environment. Agents are (or should be) disembodied bits of ‘intelligence’. Though, we will not attempt to define what intelligence is, a key attribute of any intelligent being is its ability to learn. The learning may also take the form of increased performance over time. ”

Many of the current reviews of software agents attempt to classify agents by either their ‘roles or functions (King 1995) or by their supporting technology. These approaches lead to very complex taxonomies as many agents in fixed roles, (i.e. Information retrieval agents) utilize a number of underling technologies (collaboration, mobility act.), and, conversely, many underling technologies lead to agents with many roles.

Agent developers have identified several forms of agents that are more important for large systems development. Those forms considered the most important to agent developers today are discusses below (OMG 2000):

Software agent – is an autonomous software entity that can interact with its environment. In other words, they are agents that are implemented using software. This means that they are autonomous and can react with other entities, including humans, machines, and other software agents in various environment and across various platforms.

Autonomous agent - is capable acting without direct external intervention. It has some degree of control over its internal state and actions based on its own experiences.

Adaptive agent – is capable of responding to other agents and/or its environment to some degree. More advanced from of adaptation permit an agent to modify its behavior based on its experience.

Mobile agent – able to transport itself from one environment to another.

Interactive agent – communicates with the environment and other agents.

Proxy agent – may act on behalf of someone or something, that is, acting in the interest of, as a representative of, or for the benefit of some entity.

Intelligent agent – state is formalized by knowledge (i.e., beliefs, goals, plans, assumptions) and interacts with other agents using symbolic language.

Coordinative agent – able to perform some activity in a shared environment with other agents. Activities are often coordinated via a plans, workflows, or some other process management mechanism.

Cooperative agent – able to coordinate with other agents to achieve a common purpose; nonantagonistic agents that succeed or fail together. (*collaboration* is another term used synonymously with cooperation.)

We want to pay more attention to the software agent autonomy and adaptability. When an agent has a certain independence from external control, it is considered autonomous. Autonomy is best characterized in degrees, rather than simply being present or not. To some extent, agents can operate without direct external invocation or intervention, without any autonomy, an agent would no longer be a dynamic entity. Therefore, autonomy is considered by FIPA and the OMG's Agents Working Group to be a required property of agent.

An agent is considered adaptive if it is capable of responding to other agents and/or its environment to some degree. At minimum, this means that an agent must be able to react to a simple stimulus-to make a direct, predetermined response to particular event or environmental signal. Adaptation gives to agent capacity to learn and evolve. These agents can change their behaviour based on experience.

An agent that can not respond to its environment or to other agent whose usefulness is questionable for developing agent-based systems. Adaptation as autonomy is considered by FIPA and OMG's Agent Working Group to be a required property of agent.

3. Agent Taxonomy

As we described before there are a lot of types of agent already defined. Agents may be usefully classified according to the subset of the properties that they enjoy. These properties may help us further classify agents in useful ways. The table that follows lists several of the properties mentioned above (S. Franklin 1996).

Property	**Other Names**	**Meaning**
reactive	(sensing and acting)	responds in a timely fashion to changes in the environment
autonomous		exercises control over its own actions
goal-oriented	pro-active purposeful	does not simply act in response to the environment
temporally continuous		is a continuously running process
communicative	socially able	communicates with other agents, perhaps including people
learning	adaptive	changes its behaviour based on its previous experience
mobile		able to transport itself from one machine to another
flexible		actions are not scripted
character		believable "personality" and emotional state.

Every agent satisfies the some of these properties. Adding other properties produces potentially useful classes of agents, for example, mobile, learning agents. Thus a hierarchical classification based on set inclusion occurs naturally. Mobile, learning agents are then a subclass of mobile agents.

There are, of course, other possible classifying schemes. For example, the software agents can be classified according to the tasks they perform, for example, information gathering agents or email filtering agents, or according to their control architecture. Agents may also be classified by the range and sensitivity of their senses, or by the range and effectiveness of their actions, or by how much internal state they possess (S. Franklin 1996).

As we mentioned by adding some properties to existing agent produces useful class of agent/or new type of agent. If we put agents well known properties autonomy, learning, reactions, flexibility, cooperation, and add self-adaptive behaviour, we'll develop new type of agent that is a self-adaptive (Figure 1). The agent with self-adaptive behaviour is very useful for the system that needs runtime reconfiguration.

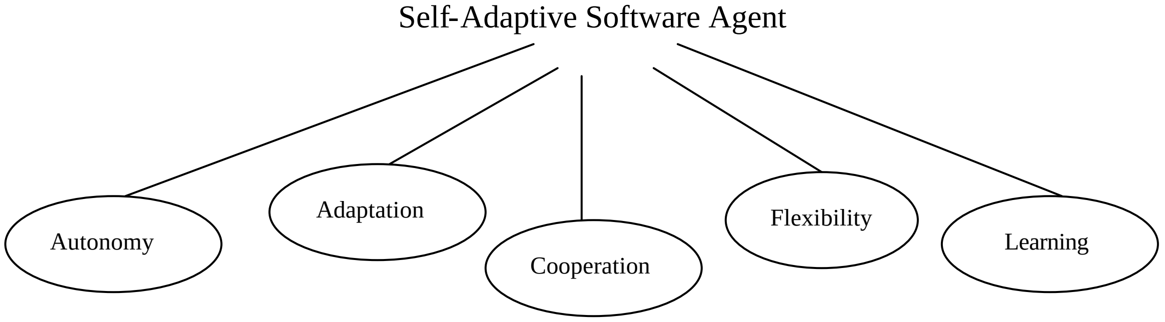

Figure 1.

4. A Self-Adaptive Software Management Approach

The Viable System Model (Beer 1981) (Beer 1985), provides a theoretically support to self-adaptive software system development (Laws 2000). Viable systems have ability to continually respond and adapt to unexpected stimuli allowing them to survive in a changing and unpredictable environment. The model identifies the necessary and sufficient communication and control systems that must exist for any organization to remain viable in a changing environment. In doing so, the model does not attempt to specify the activities that must occur in each system, instead activities are typified by a cybernetic rationale to allow either the design of activities to match the cybernetic criteria or for actual activities to be identified by their system type and hence assigned to the appropriate element of the model. Such a generalized approach allows the model to be applied to any organization regardless of size.

According the principles of the VSM and Bratman *et al.*'s [Bratman (1988)] IRMA terminology six major systems can be used to demonstrate a conceptual, architectural outline of an agent based systems.

The major systems of the Viable Systems Model

System One - *"Operations"* performs the productive operations of the organization. An organization may be composed of a number System Ones, each providing a distinct or service. Each S1 consists of an operational element controlled by a management.

System Two – *"coordination"* is concerned with coordinating the activities of S1 units. It is essentially anti-oscillatory in that it attempts to contain or minimize inter-S1 fluctuations. This is achieved by the provision of stabilizing, coordinating facilities such as scheduling and standardization information that is disseminated over all S1s, but tailored locally to suit individual S1 needs.

System Three – *"Control"* is concerned with the provision of cohesion and synergy to a set of S1units. The management processes contained within this system will be concerned with short term, immediate management issues, such as resource provision and strategic plan production, although strategic in this situation refers to planning with existing resources rather than in the normally accepted sense.

System Three* - *"Audit"* provides facilities for the intermittent audit of S1 progress and provides direct access to the physical operations of the particular S1 allowing immediate corroboration of that progress. This essentially provides additional data over and above that provided by normal reporting procedures.

System Four – *"Intelligence"* is concerned with planning the way ahead in the light of external environmental changes and internal organizational capabilities. S4 'scans' the environment for trends that may be either beneficial or detrimental to the organization and constructs developmental organizational plans accordingly. To ensure that such plans are grounded in an accurate appreciation of the current organization, the intelligence function contains an up-to-date model of organizational capability.

System Five – *"Policy"* determines the overall purpose of the organization i.e. defines the activities that are performed by S1s. as such S5 represents the policy-formulation or normative planning function. Policy formulation is informed by a "world-view" provided by S4 and models of current organizational capability populated data flowing from the lower level systems in the organization.

As this model allows to the system to have connection with environment, plan and adopt changes when it is necessary it can be used for overall configuration characteristics for both individual agent architectures and overall system architectures. In such an approach each System must be represented as an appropriate agent (e.g. Planning Agent, Learning Agent, Adaptive Agent). For communication and coordination function as a communication agent. Such an approach provides support to self-adaptive systems development.

As it was mentioned the main goal of our work is to develop agents with self-adaptive behaviour that supports run-time reconfigurable systems. VSM provides support for both agents and systems we are going to develop. Taxonomy of whole system is future work. Classification of the self-adaptive software agent is presented on Figure 2. (Laws 2000)

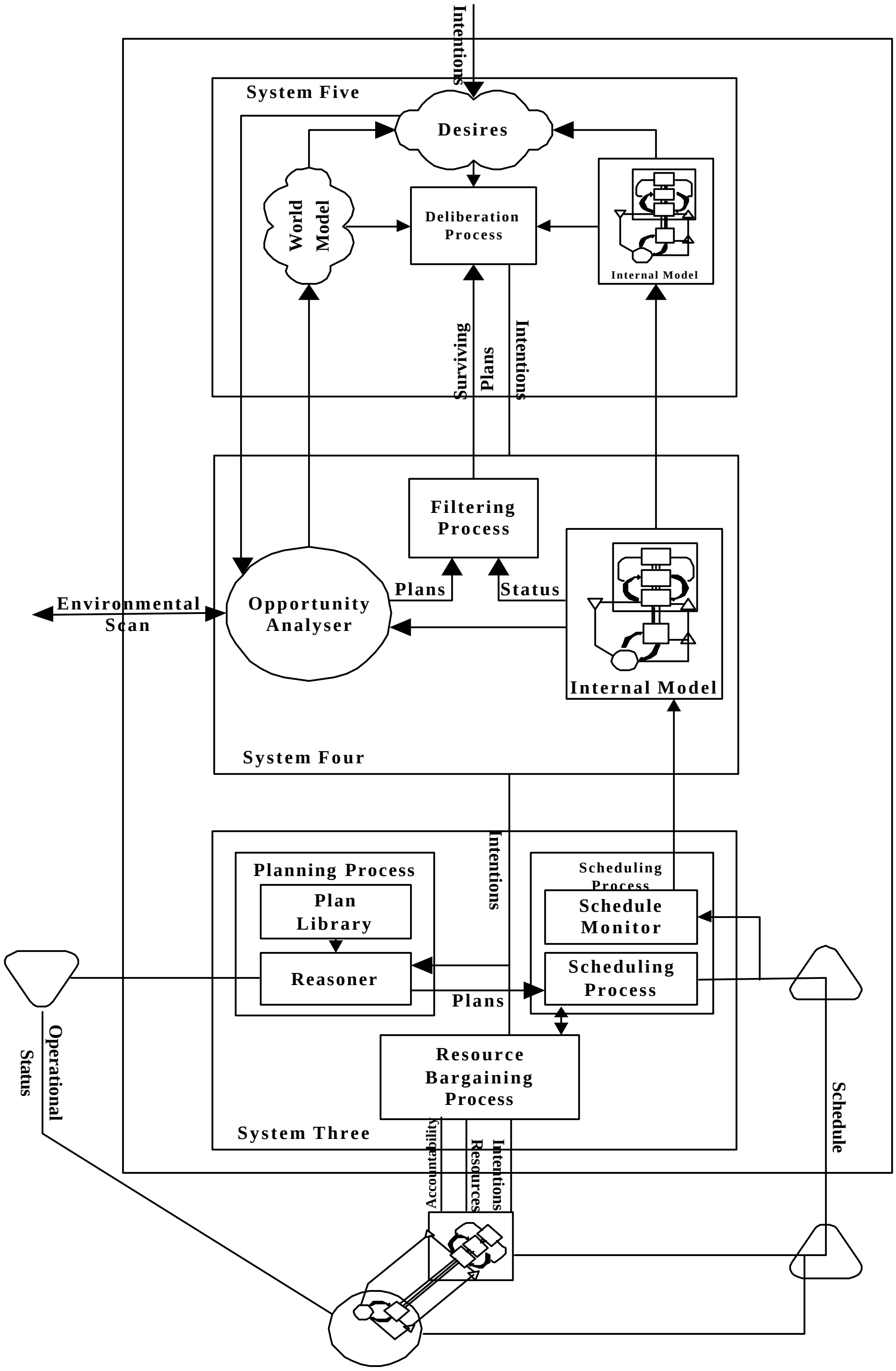

Figure 2. An outline Viable Self-Adaptive Software Agent Architecture

The architecture embeds a beliefs-desires-intentions (BDI) approach at S5 level.

System Three, using a reasoning process supported by a plan library and the capacity to audit the current status of operational System One units, structures the intentions into plans, which are then passed to a Scheduling process. The Scheduling process, in cooperation with a Resource Bargaining process, responsible for negotiating resource deployment and monitoring usage, schedule the enactment of the plan. The schedule passes to the coordinating System Two channel for dissemination to participating System One elements. The System Two channel returns schedule-monitoring information to System Three.

Environmental change is addressed in System Four, which equipped with an Opportunity Analyzer guided by System Five desires, scans the environment for detrimental events or beneficial opportunities. There are two outcomes of this process, the first is the formulation of a view of the outside world that is provided to System Five in the form of the World model. The second outcome is the production of development plans for the future of the agent, either exploiting advantageous opportunities or avoiding detrimental occurrences. To ensure that plans are founded in a realistic appreciation of the current capabilities of the agent, a model, populated by data flowing from System Three is provided. This data is further abstracted and feeds the internal view used in the deliberation process in System Five. Plans are then subjected to a filtering process to weigh between competing options and ensure compatibility with the current agent-state. Surviving plans are then passed to the deliberation process to begin the intention forming cycle again.

If we take general software agent and add the autonomous behavior we will get autonomous agent. According the described architecture of agent, on the S3 level we have autonomous agent that is capable acting without direct external intervention. On the level S4 we need to add learning behavior in order to make agent capable to interact with external world/environment, to learn the requirements the environment needs. With adding the planning behavior we will make agent to plan new changes or requirements for external world, but only on the level S5 we can make agent adaptive, adding adaptive behavior. Agent makes decision (when it is necessary) how to change existing or adapt new behavior.

5. Conclusion

In this paper we described very briefly what is the software agent and what kind of agents are developed. Presented the taxonomy of general agent with their properties and classification of self-adaptive software agent.

Discussed how it is useful to use VSM as unifying reference architecture for self-adaptive software agents and their supporting runtime reconfigurable systems.

It is clear that this project needs a lot of work to be done, but at a result of this work we are promising to create framework, that will make it easy to create intelligent software agents, with self-adaptive behaviours, with built-in capabilities for autonomous operation, monitoring their environments, reasoning, and communicating with other agents and users. Will provide tools for specifying agent behaviour and operation, tools for defining interaction between agents.

References:

Beer, S. (1981). *Brain of the Firm*, John Wiley &Sons, Chichester.

Beer, S. (1985). *Diagnosing the System for Organizations*, John Wiley and Sons, Chichester.

King, Y. A. (1995). “Intelligent Agents: Bringing Good Things to Life.” .

Laws, A. G. (2000). “Managing Complexity in Self-Adaptive Software: A Cybernetic Approach.” , S. J. Wade, ed., CMS, Liverpool John Moores University.

Nwana, H. S. (1996). “"Software Agents: An Overview" in Knowledge Engineering Review.” .

OMG. (2000). “Agent Teqnology.” , Agent Working Group, OMG.

S. Franklin, A. G. “Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents.” *The Third International Workshop on Agent Theories, Architectures, and Languages*.